DOCUMENT RESUME

ED 317 608 TM 014 719

AUTHOR De Corte, E.; And Others
TITLE Construction and Evaluation of a Powerful LOGO Learning Environment for the Acquisition and Transfer of Thinking Skills.
PUB DATE Apr 90
NOTE 45p.; Paper presented at the Annual Meeting of the American Educational Research Association (Boston, MA, April 16-20, 1990).
PUB TYPE Reports - Research/Technical (143) -- Speeches/Conference Papers (150)

EDRS PRICE MF01/PC02 Plus Postage.
DESCRIPTORS Comparative Analysis; Computer Uses in Education; *Educational Environment; *Elementary School Students; Evaluation Methods; Grade 6; Intermediate Grades; *Learning; Preadolescents; *Programing; Skill Development; *Thinking Skills; *Transfer of Training
IDENTIFIERS *LOGO Programing Language

ABSTRACT

The development of a powerful learning environment aiming at the acquisition and transfer of general thinking skills through learning to program in LOGO was studied. The following conditions were sought: (1) sufficient knowledge of the LOGO primitives and concepts; (2) mastery of the thinking skills within the programming context; and (3) explicit training for transfer. A systematic experiment was conducted in 3 6th-grade classes with 72 12-year-old students, using a pretest-posttest design with a non-treatment control group. A LOGO course, involving the training of a systematic programming strategy, was taught to two of the classes. In one of these classes, explicit instruction for transfer was also applied. At the end of the school year, the fulfillment of the conditions was tested. Findings reveal that the first two transfer conditions were fulfilled in both experimental groups; results with respect to explicit training for transfer were less positive. A series of transfer tests was administered, and data analysis revealed that transfer was obtained in both experimental groups. This finding implies that fulfillment of the first two conditions is sufficient for realizing transfer of thinking skills. Three tables present study data, and 13 figures illustrate items from the tests. (Author/SLD)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED317608

U.S. DEPARTMENT OF EDUCATION
Office of Educational Research and Improvement
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)
✓ This document has been reproduced as received from the person or organization originating it.
☐ Minor changes have been made to improve reproduction quality.
• Points of view or opinions stated in this document do not necessarily represent official OERI position or policy.

PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY
ERIK DE CORTE

TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC).

CONSTRUCTION AND EVALUATION OF A POWERFUL LOGO LEARNING ENVIRONMENT FOR THE ACQUISITION AND TRANSFER OF THINKING SKILLS

E. De Corte, L. Verschaffel, H. Schrooten,
R. Indemans, & E. Hoedemaekers

Center for Instructional Psychology
University of Leuven, Belgium

April 1990

TM 014 719

Paper presented in a Symposium on "Teaching and Learning Computer Programming", organized at the Annual Meeting of the American Educational Research Association, April 16-20, 1990.

Mailing address: E. De Corte, Center for Instructional Psychology, Department of Education, University of Leuven, Vesaliusstraat 2, B-3000 Leuven, Belgium.

BEST COPY AVAILABLE

CONSTRUCTION AND EVALUATION OF A POWERFUL LOGO LEARNING ENVIRONMENT FOR THE ACQUISITION AND TRANSFER OF THINKING SKILLS

E. De Corte, L. Verschaffel[1], H. Schrooten,
R. Indemans, and E. Hoedemaekers

Center for Instructional Psychology,
University of Leuven, Belgium

ABSTRACT

This study aimed at the development of a powerful LOGO learning environment for achieving the following conditions for the acquisition and transfer of thinking skills: 1) sufficient knowledge of the LOGO primitives and concepts; 2) mastery of the thinking skills within the programming context; and 3) explicit training for transfer. A systematic experiment was carried out in three sixth grade classes according to the pretest-posttest design with control group. A LOGO course, involving the training of a systematic programming strategy, was taught in two experimental classes. In one of these classes explicit instruction for transfer was also applied. The control group was a non-treatment group. At the end of the school year condition fulfilment was tested. The findings showed that the first two transfer conditions were fulfilled in both experimental groups; the results with respect to the third condition were less positive. Furthermore, a series of transfer tests were administered. Data analysis revealed that transfer was obtained in both experimental groups. This implies that fulfilment of the first two conditions mentioned above is sufficient for realizing transfer.

---

[1]Lieven Verschaffel is a Research Associate of the National Fund for Scientific Research Belgium

Acquiring thinking and problem-solving skills is nowadays a primary educational objective [1]. Recent research in the domain of cognitive psychology has revealed that learning to solve problems requires the integrated acquisition of three categories of abilities [2]: 1. flexible application of a well-organized domain-specific knowledge base, involving concepts, rules, principles, formulas, and algorithms; 2. heuristic methods, i.e. systematic search strategies for problem analysis and transformation; 3. metacognitive skills, involving knowledge concerning one's own cognitive functioning on the one hand, and the self-monitoring and regulation of one's own cognitive processes on the other. Learning to solve problems also supposes that the learned knowledge and skills can be applied in new problem situations, even in other content domains; in other words, it assumes that transfer will occur.

In this respect, it has often been claimed that learning to program offers great potential for the acquisition and transfer of important heuristics and metacognitive skills. A major advocate of this "cognitive effects hypothesis" with respect to computer programming is Papert. In his book Mindstorms: children, computers and powerful ideas [3], he argues that LOGO leads to the development in children of general thinking and problem-solving skills such as planning, problem decomposition, and debugging on the one hand, and to the acquisition of powerful concepts having a wide application, such as variable and recursion on the other. Another key idea of Papert's theory is that LOGO should be acquired according to a self-discovery strategy, analogous to how a young child learns to speak.

Feurzeig, Horowitz and Nickerson [4] have given a more systematic overview of the cognitive skills that children could acquire while learning to program. The most important ones mentioned by these authors are:

1. rigorous thinking and the ability to express one's thoughts accurately and precisely;
2. understanding and being able to apply important general concepts such as procedure, variable, function, recursion;
3. mastery of heuristic methods such as planning, decomposing a problem into its constituent parts and thinking of an analogous problem;
4. the ability to discover and debug errors in a solution procedure,
5. awareness that for most problems there are different solution strategies.

One will notice that this list contains aspects belonging to each of the three previously mentioned categories of skills that a competent problem solver masters.

However, an important question is whether there is empirical evidence supporting the cognitive effects hypothesis. A review of the literature shows that this hypothesis is mostly justified on the basis of a rational analysis of programming, according to which this complex activity requires a number of skills that are considered to be of importance in learning, thinking and problem solving in general. Nickerson [5, p. 42] writes in this respect: "Perhaps the basic reason for the belief that programming might be an effective vehicle for the acquisition of generally useful cognitive skills is the assumption that programming is prototypical of many cognitively demanding tasks. It is a creative endeavor requiring planning, precision in the use of language, the generation and testing of hypotheses, the ability to identify action sequences that will realize specified objectives, careful attention to detail and a variety of other skills that seem to reflect what thinking is all about". Although such considerations are quite interesting, they do not offer a sufficient basis for accepting this hypothesis.

Convincing scientific evidence is needed, the more so since studies in other domains showed that transfer is difficult to achieve (e.g. [6, 7]), and the results of a number of pioneering investigations with respect to programming carried out over the first part of the past decade also did not support this cognitive effects hypothesis. However, the latter sobering outcomes can be attributed to the following two aspects of these investigations. First, the pupils did not acquire sufficient programming ability, due to the short duration of the hands-on experience (on the average about 25 to 30 hours) on the one hand, and to the absence of systematic instruction on the other. Second, an explicit and systematic orientation towards transfer was lacking (for a review, see [8, 9]).

Since 1985 we have been engaged in a research project "Computers and Thinking" that relates to the cognitive effects of learning to program in LOGO. The major objective was the development, implementation and evaluation of a powerful learning environment aiming at the acquisition and transfer of general thinking skills. We focused on a subset of the problem-solving skills that are expected to be influenced by programming experience: two metacognitive skills

(planning and debugging), and two heuristics (problem decomposition and construction of an external problem representation).

Taking into account the results of earlier work about the cognitive effects hypothesis on the one hand, and the recent cognitive psychological literature on the other, we hypothesized that fulfilment of the following three conditions is crucial in order to attain transfer of cognitive skills:

1. the pupils have acquired sufficient domain-specific knowledge (i.e. LOGO language features and concepts);
2. they have achieved mastery of the heuristics and metacognitive skills within the LOGO environment;
3. they have learned how to apply the skills taught in the programming environment in at least one other content domain.

As argued before, previous research has convincingly shown that fulfilment of the first condition alone is not sufficient in order to obtain transfer of thinking skills. Therefore, the focus of the present study was the necessity of the transfer conditions 2 and 3. The general hypothesis was formulated as follows: "If the first two conditions are fulfilled, transfer of thinking skills will occur; fulfilment of the third condition will enhance the transfer effect".

The project was carried out in two stages. During the school year 1986-87 an exploratory study was undertaken, aiming at the development, tryout and revision of a LOGO teaching-learning environment, and at the construction of instruments for measuring pupils' knowledge and skills within the LOGO environment (transfer conditions 1 and 2) (see [10] for more detailed information). To test the hypothesis mentioned above, a more systematic teaching experiment was undertaken during the school year 1987-88. In that experiment the teaching-learning environment was extended with one crucial component, namely the explicit training for transfer (transfer condition 3). The present article deals with the design and the results of this second study.

METHOD

Subjects

The subjects were 12 year-old children, who had no prior experience with computer in general and with LOGO in particular. All participants were drawn

from three sixth-grade classes, with 24 pupils each (n=72). Two classes served as experimental groups (E1 and E2), while the third one was a control group (C). In the beginning of the school year an intelligence test and a school achievement test were administered in all three classes; no significant differences between the three groups were found.

Design

The experiment was carried out according to the pretest-posttest design with control group. In E1, the fulfilment of the first two transfer conditions mentioned above was pursued through a 60-hour LOGO course, involving the systematic instruction of the primitives of the LOGO language on the one hand, and of a strategy to write programs using LOGO's graphical mode on the other. This strategy involves two main phases, namely a planning phase and an executing-and-testing phase. It constitutes the operationalisation of the previously mentioned general thinking skills that we aimed at in the LOGO course: planning, debugging, decomposition, and external representation. In E2, all three transfer conditions were pursued: on top of the LOGO course, the children were taught how to apply the learned skills in another context, namely in solving multi-step mathematical word problems. The control class was a non-treatment group.

LOGO learning environment

A LOGO course outlined in Table 1 was taught in the two experimental groups one afternoon each week during the whole school year (approximately 60 hours).

Insert Table 1 here

The teaching was taken care of by members of the research team, in cooperation with the teacher. The computer room, which was directly accessible form the regular classroom, was equipped with nine Philips MSX-microcomputers. The course consisted of two major components, namely the teaching of the LOGO primitives and concepts (transfer condition 1) on the one hand, and the training of the programming strategy (transfer conditon 2) on the other.

Teaching LOGO primitives and concepts (transfer condition 1) - The content of the LOGO course was limited to the so-called turtle graphics. Moreover, the children were only taught those LOGO primitives and concepts considered necessary with a view to the acquisition of the intended programming strategy. More specifically, the following notions of LOGO were treated: making simple drawings using the basic LOGO primitives (FD/BK, RT/LT, PU/PD), working with the REPEAT command, writing procedures and superprocedures.

Starting from a moderate constructivist conception [11], we assumed that a powerful learning environment is characterized by a good balance between discovery learning on the one hand, and systematic instruction on the other.

A first example of structuring the pupils' learning processes is presenting the LOGO primitives and concepts in a well-considered sequence. We only passed on to the next primitive or concept if the results of an intermediary test showed that the former was sufficiently mastered.

Second, the teaching of a new LOGO primitive or concept was performed in a rather systematic way, mostly via demonstration at the computer or in a discussion format. During these demonstrations and discussions we made use of our knowledge of children's typical errors and misconceptions, that we had acquired during the exploratory study. For example, to avoid the typical misconceptions and errors with respect to angles [10], we inserted activities like playing turtle, indicating the turning angles in a clear and uniform way, estimating angles, and measuring angles using a protractor.

Afterwards, the children were given ample opportunity to practise the newly learned primitive or concept mostly in small groups of two to three pupils at the computer. To stimulate exploratory activities, challenging tasks were provided. We also constructed game-like microworlds, in which the children could exercise certain LOGO notions in an intensive but attractive manner. The "direction game", for instance, aimed at practising the primitives LT and RT. Its goal was to send the turtle, that moved with a certain speed and in a certain direction over the screen, to its nest, using LT and RT instructions. The practising phase was concluded with a discussion of children's difficulties, findings and strategies.

To wind up with, the children each time had the occasion to practise the newly mastered LOGO knowledge in integration with the previously learned ones in more extensive self-chosen projects.

Teaching of a strategy for writing LOGO programs (transfer condition 2) - The LOGO programming strategy that was taught, consists of two main phases: a planning phase, and an integrated executing-and-testing phase. As said before, this strategy involved two metacognitive components, namely planning and debugging, and two heuristics, namely problem decomposition and construction of an external representation. Planning and debugging were instructed respectively in the first and the second phase. The two heuristics were taught in the planning phase.

In the planning phase, carried out independently from the computer, three steps are distinguished (see Figure 1).

Insert Figure 1 here

- Making a drawing of the intended screen effect.
- Constructing a tree-like diagram in which the complex drawing is subdivided in building blocks that are easy to program; this diagram involves at the same time the sequence in which the different parts have to be drawn on the screen.
- Making separate drawings of the different building blocks or parts, indicating for each part the lengths and angles as well as the start and the end position of the turtle.

Once the planning is completed, the integrated executing-and-testing phase on the machine can begin. This activity is guided by two principles:

- Top-down programming, involving that the children are taught to start with the most global procedure, called the "mother procedure", which consists of the names of the subsequent parts from the second level of the tree-like diagram, together with the names of the "connecting links" that move the turtle from the end position of a previous part to the start position of the next building block[2]; subsequently each component of this procedure is specified until the lowest level of the tree-like diagram is reached.

---

[2] In the present study, it was agreed that the name for a connecting link between two building blocks consists of the following three parts: the characters "CL" (from Connecting Link), followed by a dash and the first characters of the former and the following block (e.g. connecting link between WINDOW and MOTOR: CL-WM).

- Immediate testing and debugging of each new procedure after defining it. By calling the "mother procedure", the result appears on the screen and can instantly be evaluated; furthermore the error message ("There is no procedure named...") indicates which procedure has to be written next.

To illustrate these principles, we present the interaction with the computer of a child who applies the strategy for drawing the car in Figure 1. In accordance with the principle of top-down programming, the child first defines the mother procedure (CAR):

```
TO CAR
  WINDOW CL-WM MOTOR CL-MW WHEELS
  END
```

[3]

Then, following the second principle, the child types the name of the mother procedure (CAR), resulting in the error message "THERE IS NO PROCEDURE NAMED WINDOW AT LEVEL 1 OF CAR". Next the WINDOW procedure is defined:

```
TO WINDOW
  REPEAT 2 (FD 15 RT 90 FD 40 RT 90)
  END
```

As a result of the subsequent testing (CAR), the following screen effect is obtained (see Figure 2):

Insert Figure 2 here

The error message at the bottom of the screen indicates that the connecting link between the window and the motor (CL-WM) is the next procedure to be written. The child types:

```
TO CL-WM
  PU RT 90 BK 10 PD
  END
```

Calling the mother procedure (CAR) results in the following screen effect (see Figure 3):

---

[3]For the sake of clarity, the children were taught to start a new line for each procedure in a superprocedure. However, in this text all procedures are on one line because of the limited space available.

Insert Figure 3 here

Subsequently, the child writes the procedure for the motor; e.g.

```
TO MOTOR
  COVER CL-CL LIGHTS CL-LR RADIATOR
  END
```

When afterwards the mother procedure (CAR) is called, the turtle again draws the window and the connecting link between the window and the engine, and then gives the message "THERE IS NO PROCEDURE NAMED COVER AT LEVEL 1 OF MOTOR". The next procedure to be written is the COVER procedure.

This process is repeated until all building blocks and connecting links of the MOTOR procedure, and subsequently all parts of the remaining components of the mother procedure (CAR) are defined.

The explicit teaching of this programming strategy constituted the major component of the LOGO course. We already tried out our learning environment in the exploratory study [10]. This lead to the following findings. When asked to write a program for a complex drawing at the end of the course, several children did not construct a tree diagram spontaneously. The interviews revealed that a number of them did not see the benefits of using the strategy that was taught. Other pupils expressed their negative feelings towards the taut strategy. However, we also found that several children, if explicitly asked to do so, were unable to construct a correct tree-diagram. Therefore, we modified the teaching-learning environment: in the present study the intended strategy was taught in a more systematic manner, however without being too strict; furthermore we tried to motivate the children to use the strategy. In that perspective we applied several teaching strategies that are considered in the recent literature to be effective for the instruction of general thinking skills: modelling, scaffolding, coaching, fading, articulation, reflection and exploration [12].

At first, the entire strategy was demonstrated by an "expert" (a member of the research team), with a view to helping the children to build a conceptual model of the steps and processes required to carry out the task (modelling).

Then each component of the strategy was treated and practised separately. In this phase scaffolding was used, involving that the teacher provides direct

support to the pupils in carrying out those parts of the task that were beyond their unassisted efforts. For instance, a computer program was used to help the children in constructing a tree-diagram. This program guided the decomposition process during the planning phase through a series of simple questions (e.g. Is MOTOR simple enough? Which are the constituting parts of MOTOR?). Through the repeated use of this program for constructing diagrams for a diversity of drawings, we pursued the internalization of the planning strategy.

Next, the children were given ample opportunity to practice the whole strategy, working on progressively more difficult problems in small groups of two to three pupils (exploration). In order to help the children perform the strategy autonomously, they were provided with several charts containing the different steps of the strategy and other useful information. In the beginning the pupils were guided intensively (coaching), using hints, explanation and feedback. In doing so, we took into consideration the errors, misconceptions and difficulties observed in the exploratory study. For instance, the criteria for a correct tree-diagram and the classification of errors derived from an analysis of the material of the previous exploratory investigation, were made explicit to the pupils, and this terminology was used while discussing the tree-diagrams that were made in all projects throughout the school year. Later on teacher help and supervision decreased with increasing student ability (fading).

Throughout the instruction and practice of the strategy, the children were stimulated to spell out the procedures and skills they used (articulation), and to compare them with those of other children and with the expert model (reflection). With a view to attaining transfer, the explicit and intentional abstraction of the trained thinking skills was pursued.

To enhance children's motivation to use the programming strategy, a substantial amount of time was spent demonstrating its use and benefits. For instance, programs were written for a series of similar, but not identical drawings; this revealed the benefits of modular and planful working methods, such as reusability, compartmentalization, and efficiency. Furthermore, it was demonstrated that the actual coding of the program at the computer proceeds far more easily and efficiently when one can start from a complete plan.

Finally, to avoid that the pupils developed an aversion for a too rigidly and tautly imposed strategy, we allowed them to follow the procedure less strictly as the school year proceeded. This is perfectly in accordance with our

view of the learning process. We consider the full strategy as an appropriate initial structure of action, in which the solution process is externalized very extensively. However, during the further course of the acquisition process, this structure of action can progressively be abbreviated (by omitting some parts) and internalized.

Explicit instruction for transfer (transfer condition 3)

In the second experimental group (E2) explicit instruction for transfer was provided in an additional mini-course in word-problem solving, in which the pupils learned to apply the skills acquired in the LOGO context (namely planning, debugging, decomposition, and external representation) in solving multi-step arithmetic word problems. More specifically, the children were taught a strategy involving two main phases: a planning phase, and an executing-and-testing phase. As in LOGO, the planning phase consists in subdividing the multi-step problem into subproblems that can be solved easily; this activity can also be represented in a tree-like diagram (see Figure 4). In the integrated executing-and-testing phase, each subproblem is solved and the outcome is immediately tested.

Insert Figure 4 here

Mastery tests

Before assessing the transfer effect, we examined fulfilment of the transfer conditions mentioned above. In both experimental classes the realization of the first two conditions was assessed one month before the end of the course, using one or more mastery tests for each condition. For each test a mastery criterion for the whole class was put forward: 75% of the children had to obtain 2/3 of the maximum score on the test. When this criterion was not attained, we considered the relating condition as not sufficiently fulfilled. In that case, remedial instruction was provided for those children who did not meet the predetermined criterion. Afterwards, condition fulfilment was reassessed at the end of the school year. The tests with respect to word-problem solving (third transfer condition) were

administered only once, because there was no time available for remedial instruction and retesting.

Mastery of the LOGO knowledge (transfer condition 1) - The test to assess the knowledge and understanding of the LOGO primitives and concepts consisted of 20 items relating to the following topics: the elementary primitives (FD/BK, RT/LT, PU/PD), the REPEAT command, and the flow of control in superprocedures. The children either had to indicate which of four alternative LOGO commands or procedures would produce a given screen effect, or to select the appropriate drawing for a particular LOGO command or program. The choice of the alternatives was inspired by the common errors children make. Figure 5 contains an example of an item for each of the three topics.

Insert Figure 5 here

For each correct answer, one point was given. A correction for guessing was applied on the total score using the following formula: T = C - (I / A - 1) (Total score after correction, Correct answers, Incorrect answers, and Alternatives).

Mastery of the programming strategy (transfer condition 2) - To assess mastery of the programming strategy, three tests were administered. Two paper-and-pencil tests dealt with the construction of a tree-like diagram (= planning stage); the third one was an individual test at the computer, and concerned the integrated executing-and-testing phase.

In the first planning test each item consisted of a drawing with two tree-like diagrams, only one of which was correct. The children were asked to indicate the correct tree diagram for the drawing. Figure 6 shows an example of an item.

Insert Figure 6 here

The errors included in the tree-diagrams were again selected on the basis of children's solutions on tests administered in the exploratory study [10]. The error analysis of these tasks resulted in a classification in severe and slight errors; in each category several types of errors are distinguished. A severe

error is one that results in a faulty screen effect, e.g. one part of the drawing is forgotten in the tree-diagram. A tree-diagram containing a slight error deviates from the expert model, but this does not lead to an erroneous screen effect, e.g. two identical parts are given a different name. Half of the 20 items in the test contained a severe mistake; the other half a slight one. A correct solution on an item of the first category yielded two points; the other ones one point. Hence the maximum score on this test was 30.

In the second planning test the children had to construct autonomously an appropriate tree-like diagram for a given drawing (see Figure 7).

Insert Figure 7 here

In scoring this test, we started from an expert model of the diagram. When the answer of the pupil corresponded to this diagram, the maximum score, namely 10, was attributed. For slight and severe faults, one and two points were subtracted respectively.

The individual test concerning the integrated executing-and-testing phase was administered at the computer. The child was asked to write a LOGO program for a relatively complex drawing starting from a given plan (see Figure 8). To enable us to analyse children's debugging activities, four errors were deliberately included in this plan.

Insert Figure 8 here

During the test, the teacher sat beside the pupil and intervened every time the child deviated from the executing-and-testing strategy as taught in the LOGO course. First, the teacher asked a well-chosen question (e.g. Is this the next procedure that has to be defined?). When the pupil was incapable to answer this question, the teacher responded himself (e.g. No, the error message indicates that the "BRIDGE" procedure is the next procedure to be written). Every intervention was registered; two points were given when no intervention was needed, one point when the teacher had to ask a question, and zero points when he also had to provide the answer.

Word-problem test (transfer condition 3) - To assess mastery of the strategy for solving multi-step word problems (the third transfer condition) in

E2, analogous tests as for LOGO were used. Two tests related to the planning phase. In the first one the children had to indicate the best of two given tree-like diagrams for a series of 10 word problems. Again, part of the items (namely 4) contained a severe mistake; the other part a slight one (namely 6). An error classification scheme was developed in analogy with the one used in LOGO. For a correct solution of an item containing a severe mistake, two points were given; for slight errors one point.

In the second test the children were asked to construct autonomously appropriate tree-like diagrams for five given word problems. In scoring this test we started from an expert model of the diagram. When the answer of the pupil corresponded to this diagram, the maximum score, namely 10, was attributed. For severe faults, two points were subtracted, for slight faults one point.

The third test dealt with the integrated executing-and-testing phase. The task was to solve a number of problems starting from given diagrams. Each correctly solved element in the diagram yielded one point.

Transfer tests

The children of all three classes were given a series of transfer tasks in order to evaluate their ability to apply the thinking skills trained in E1 and E2 to other situations. At the beginning of the school year, a battery of three tests was administered, namely the "mazes" test, the "error detection" test, and the "blocks" test. At the end of the course, two tests were added to this battery, namely the "weekdays" test and the "boxes" test.

Mazes test - This test, a computer test for planning behavior, was administered individually at the machine. The computer subsequently offered three mazes, containing a hungry beatle and three salads (see Figure 9). The task consisted in sending the beatle by the shortest road through the maze to the three salads. The third item, however, was unsolvable because one of the salads was unattainable.

Insert Figure 9 here

This test was developed to measure planning skill. A first measure for planning ability, "mazes/pl1"[4], related to the first two mazes. The score was based on the total distance the beatle covered. Planning was required to find the best sequence. Consequently, the shorter the covered distance was, the better the result. The second planning measure, "mazes/pl2", concerned the third maze. We registered at which moment the pupil discovered that this item was unsolvable. Pupils who made this discovery before giving the first command obtained two points; we assumed that they made a plan beforehand. Children who saw the unsolvability only after the beatle had eaten the first two salads, were given one point, and those who tried to send the beatle to the last salad, got zero points.

Error detection test - This test, which was inspired by the work of McCoy Carver [13], contained three stories, each one telling about person A giving person B directions. Person B followed the directions perfectly, but the obtained result was not as intended because one of the instructions was wrong. The task was to find and to fix the bug in the directions, so that the anticipated result would be obtained (see Figure 10).

Insert Figure 10 here

This test was administered individually. It measures two thinking skills, namely planning and debugging. First, the bug had to be located in the list of instructions; a planful search of the bug is based on a preliminary comparison of the discrepancy between the obtained and the intended effect. When such a plan was made before starting to execute the directions one by one, one point for planning skill ("error/pln") was given; a child obtained zero points if this phase was skipped. Subsequently, to correct the bug, the directions had to be executed in order to compare the result with the intended outcome. One point

---

[4]The names of the transfer measures are composed of two parts: the first part refers to the concerned transfer test; the second part to the thinking skill that is being measured (planning, debugging, problem decomposition, and construction of an external representation).

for debugging ability ("error/deb") was attributed if the bug was appropriately corrected, zero points when this was not the case.

Blocks test - In this test, one had to describe 10 complex figures in terms of their component parts, making use of a series of given simple elements or other less complex figures (see Figure 11). To describe the figures, the least possible elements had to be used.

Insert Figure 11 here

Solving this test efficiently requires application of the decomposition heuristic. The scores for this skill ("block/dec") ranged from 1 to 10; one point was given for each correctly solved item.

Weekdays test - This test consisted of ten items such as "If tomorrow is Saturday, which day is the day before yesterday?" [14]. An efficient strategy to solve such tasks consists in the use of an external representation. The score for this heuristic ("weekd/ext") was one or zero: one point for a correct representation, zero points in all other cases. This test also measures the decomposition skill. In order to obtain the correct solution, it is necessary to decompose the problem, to solve each part separately and to link up the answers to a conclusion [14]. The score for "weekd/dec" was one if the external representation contained a decomposition of the problem; zero if this was not the case. Figure 12 shows two examples of an external representation and a decomposition of the problem mentioned above.

Insert Figure 12 here

Boxes test - Children's mastery of the use of external representations was also tested in the boxes test, consisting of ten items; for example, "There is one big box. Inside the big box there are two separate medium boxes. Inside one of the medium boxes there are three small boxes. Inside the other one there are four boxes. How many boxes are there all together?" [14]. One point was given if an appropriate graphic representation of the problem was made ("boxes/ext");

otherwise, the child obtained zero points. Figure 13 contains two examples of an external representation for this item.

Insert Figure 13 here

RESULTS

Results on the mastery tests

Table 2 shows the number of pupils in E1 and E2 that reached the predetermined mastery criterion (2/3 of the maximum score) on the different tests by the end of the school year.

Insert Table 2 here

The results on the LOGO knowledge test indicated that by the end of the experiment most children in both experimental classes (21 and 17 in E1 and E2 respectively) had acquired a sufficient level of knowledge and understanding of the LOGO primitives and concepts (first transfer condition).

The outcomes on the measures relating to the programming strategy (second transfer condition) were also very positive. The results on the first planning test showed that in E1 and E2 resp. 19 and 23 children were sufficiently able to indicate the correct diagram for a series of drawings. The second planning test revealed that in E1 and E2 resp. 20 and 19 pupils succeeded in constructing autonomously an appropriate tree-like diagram for a given drawing. On the test concerning the executing-and-testing stage of the programming strategy, namely writing a LOGO program for a relatively complex drawing starting from a given plan, most of the children (20 in E1 and 22 in E2) were able to apply the instructed strategy efficiently: they used the top-down programming style fairly consistent; each time a new procedure was defined, it was tested by calling the so-called mother-procedure; and when an incorrect screen effect was obtained, detection, analysis, and debugging of the programming error was performed in a systematic and straightforward way. The preceding results justify the conclusion that the children were able to apply the intended general thinking skills within the LOGO environment. In summary,

by the end of the LOGO course both the first and the second transfer condition were fulfilled.

Table 2 also shows that the results on the tests with respect to the third transfer condition, namely explicit orientation toward transfer to other domains, were less positive. In E2, only 10 out of 24 children were able to apply autonomously the planning strategy in the domain of arithmetic word problems (second planning test). Although their performance on the other two tests was quite good, we consider the third condition as not sufficiently fulfilled, the argument being that the second planning test was undoubtedly the most critical measure of strategy mastery. Unfortunately, because the school year was at its end, there was no time available for remedial instruction. Taking into account this latter finding, a conclusive test of the second part of our main hypothesis, namely that the fulfilment of the third condition enhances transfer, became impossible in the present study.

Results on the transfer tests

A multivariate analysis of variance (MANOVA) was applied to analyse the data for each of the four thinking skills separately. When all transfer measures were involved in the pretests as well as in the posttests, a two-way MANOVA was performed (groups x testsessions); in the other case we applied a one-way MANOVA (groups). When the multivariate analysis for a particular skill showed a significant effect, an addidional univariate analysis (ANOVA) was conducted to specify this result. More specifically, for each of the transfer measures related to this skill we computed the size and the direction of the effect.

Separate analyses were performed for E1 and C on the one hand, and E1 and E2 on the other. According to our initial hypothesis, E1 would make more progress from the pretests to the posttests than C; moreover E2 would outperform E1.

The multivariate analyses showed that the E1 group differed significantly from the C group on problem decomposition, construction of an external representation, and debugging (see Table 3). The additional univariate analyses indicated significant effects on each of the transfer measures with respect ot the three skills; moreover these effects were in the expected direction: E1 significantly outperformed C on error/deb ($p=.001$), block/dec ($p=.008$),

weekd/dec (p=.005), boxes/ext (p<.001) and weekd/ext (p=.001). This leads to the conclusion that the LOGO treatment was successful in obtaining transfer for three out of the four thinking skills that were taught.

Insert Table 3 here

The MANOVA on the measures for planning skill (mazes/pl1, mazes/pl2, error/pln) showed no significant differences between E1 and C: both groups performed equally well on the tests requiring this skill. This is somewhat surprising, since the planning component was the core component of the trained strategy.

As we expected on the basis of the non-fulfilment of the third transfer condition, E2 did not make more progress than E1. Unexpectedly, we even found significant effects in the opposite direction: E1 outperformed E2 on the measures for problem decomposition, construction of an external representation, and debugging (see Table 3).

DISCUSSION

In the present acticle we described the design and the results of a one-year teaching experiment, in which we implemented and evaluated a LOGO teaching learning environment aimed at the acquisition and transfer of general thinking skills. The study took place in three sixth grade classes according to the pretest-posttest design with control group.

Three conditions that are important with a view to the attainment of transfer were derived from the literature. The first two conditions - acquisition of a sufficient level of domain-specific knowledge on the one hand, and mastery of the intended thinking skills within the LOGO context on the other - were pursued in a 60-hour LOGO course taught in both E1 and E2. In E2 explicit training for transfer (third condition) was also applied in a supplementary math course.

Before assessing the transfer effect, we examined fulfilment of the transfer conditions. The results on the LOGO mastery tests indicated that the first two conditions were achieved: the pupils had acquired a sufficient level of LOGO knowledge and could successfully apply the thinking skills in the LOGO context. In our opinion, these positive results can be attributed to the carefully constructed instructional environment, in which the shortcomings of

former studies were overcome. Two important characteristics of this learning environment were: 1) its time-intensive character, and 2) the balance between exploratory learning activities on the one hand, and systematic instruction (e.g. the explicit teaching of a programming strategy) on the other. However, in evaluating these findings appropriately, one should certainly take into account that the results were obtained in a rather a-typical situation (60 hours of LOGO instruction, 9 computers and 4 experienced teachers for 20 children), which is not representative of the current situation in Belgian elementary schools.

We did not succeed in realizing the third transfer condition. After several lessons focussing on the explicit instruction in applying the problem-solving skills taught in the LOGO environment to the domain of mathematical word problems, most children in E2 were not sufficiently able to do so at the end of the course. This disappointing finding is probably due to the existence in children of inefficient strategies, attitudes and beliefs about word problem solving, which they had acquired over the years during their normal math lessons, and which were difficult, if not impossible to change substantially within the available amount of time [15].

The transfer results with respect to the first part of the hypothesis, namely that fulfilment of the first two transfer conditions is sufficient for obtaining transferable thinking skills, are rather encouraging. Generally speaking, the LOGO group (E1) significantly outperformed the control group on the transfer tests for three skills, namely problem decomposition, use of an external representation, and debugging. Similar positive transfer results were reported in other recent studies (for example [13, 16]; for an overview, see [8, 9]). A common factor in these successful investigations, is the explicit instruction involving "mindful abstraction" [6, 17] of the intended general thinking skills within the LOGO environment.

The second part of the hypothesis, namely that decontextualisation by teaching the skills in other domains (condition 3) would enhance the transfer effect, was not confirmed. The pupils who followed the math course (E2) did not obtain better transfer results than those from E1; we even found that in E2 less transfer occurred. However, in our opinion it would be premature to conclude from these findings that explicit training for transfer has no or even a negative influence on the development of general thinking skills. Indeed, the absence of the expected transfer effects, can be explained by the

non-fulfilment of the third transfer condition. Hence, a conclusive test of the importance or the necessity of the third transfer condition became impossible in this study. On the other hand it is often suggested in the transfer literature that the application of the intended thinking skills in a diversity of situations is crucial to bring about abstraction and decontextualisation (see also [6, 18]). Therefore, we tend to hold on the hypothesis that if one succeeds in realizing the explicit instruction for transfer properly, additional transfer effects will occur.

A plausible explanation for the fact that in E2 less transfer was found, relates to motivational factors and classroom atmosphere. Although systematic data are lacking, there are several indications that the replacement of the LOGO course by the math course in E2 resulted in a decrease in the positive attitudes of the children of that class.

The starting point of the present study was the question "Is learning to program a vehicle for the development of thinking skills in elementary school children?". On the basis of the obtained results and the gathered experience, we think we can formulate a moderately positive answer to this question. Yes, the findings suggest that learning to program can result in the acquisition and transfer of general thinking skills. However, the positive effects do not merely result from the unique characteristics of the LOGO language as such, but rather from the qualities of the instructional system that is applied in teaching programming. In other words, LOGO in itself is not a vehicle for thinking, but it can be a useful device for the acquisition of general thinking skills, if it is embedded in a powerful teaching-learning environment, i.e. a context that provokes in children the learning processes necessary to reach the intended goals.

Future research should aim at the identification and analysis of the critical aspects and dimensions of such powerful environments. In that perspective, we refer briefly to some research-based ideas from the domain of instructional psychology that should be taken into account. As a frame of reference we use Resnick's [19] distinction between three major components in a theory of learning from instruction, namely a theory of expertise describing skilled performance in a domain, an acquisition theory explaining the processes of learning and development necessary to achieve expert performance, and a theory of intervention describing appropriate teaching methods and

instructional strategies for eliciting those processes of learning and development.

So far cognitive psychological research has mainly focused on the development of a theory of expertise, resulting in a better understanding of the nature of expert problem-solving and thinking processes. This work led amoung others to the identification of three aforementioned major categories of skills that are involved in expert problem solving, namely domain-specific knowledge, heuristic methods and metacognitive knowledge and skills.

Although there is by now no well elaborated acquisition theory available, recent research has already yielded some important characteristics of learning processes that should be taken into account in designing learning environments. Major aspects in this respect are: the constructive nature of learning, meaning that learners are not passive recipients of information, but that they actively construct their knowledge and skills through interaction with the environment, and through reorganisation of their own mental structures; the role of prior knowledge in general, and of children's informal knowledge and skills in particular, as a starting point for the acquisition process; the need to anchor learning in real life experience and to provide learning tasks that are representative of the multiple situations in which knowledge will have to be used later on; the progressive internalization and formalization as a guiding principle for acquiring new knowledge and skills; the influence of beliefs and of motivation on learning; and the importance of the explicit teaching for transfer (for more details, see [20]).

An interesting contribution to the intervention component of a theory of learning from instruction is the recent work of Collins, et al. [12] to which we already referred before. Starting from the cognitive apprenticeship view of teaching and learning which embeds the acquisition of knowledge and skills in the social and functional context of their use, the authors identified four dimensions that are relevant in designing learning environments, namely content, method, sequence, and sociology.

With respect to content, an ideal learning environment should focus on the acquisition of all categories of skills that experts master. In addition to the three categories discussed in the first section of this paper - domain-specific knowledge, heuristic methods, and metacognitive strategies - Collins et al. [12] mention a fourth type of skills that experts apply, namely learning

strategies, i.e. strategies for acquiring any of the three other types of content.

With a view to helping students to acquire and integrate those different categories of knowledge and skills the teacher can apply six different methods falling roughly into three categories. The first three - modelling, coaching and scaffolding - aim at helping the pupils acquire an integrated set of cognitive and metacognitive skills via observation, guided and supported practice, and feedback. Two other methods - articulation and reflection - are techniques that help pupils analyse their own cognitive and metacognitive activities and confront them with the ideas and strategies of other learners, and ultimately, with a mental model of expert performance. Exploration, finally, intends to increase the learner's autonomy in skilled problem solving as well as in discovering, identifying, and defining a new problem.

With respect to the sequencing of learning tasks, two principles are mentioned: 1. progressive complexity and diversity, such that competent performance requires more and more of the domain-specific knowledge as well as a larger variety of cognitive and metacognitive skills; and 2. global before local skills, involving that the orientation towards the complex task as a whole should precede the practising of partial, lower-level tasks.

Finally, the authors describe a series of guidelines that are important with a view to realizing a favorable social context for learning: 1. situated learning involving that students should be given tasks and problems representing the diversity of situations to which they will have to apply their knowledge and skills afterwards; 2. organizing opportunities for contact with and observation of experts; 3. enhancing intrinsic motivation for learning; 4. fostering cooperative learning through small group problem solving; 5. organizing classroom dialogues aiming at the identification, analysis, and discussion of students' problem-solving strategies and processes.

Although we developed our LOGO learning environment before taking cognizance of this model, the resemblance is striking. Indeed, in our instructional system we used most of the above mentioned intervention techniques. Also, most of the principles concerning the sequencing of tasks and the social context of learning were taken into consideration.

We now pass onto an interesting issue relating to the construction of such powerful learning environments, especially when it concerns the training of thinking skills through computer programming in particular, but also through

educational software in general. The major conclusion that we derived from the available research, is that the totally open contact with a programming language is not sufficient for the acquisition and transfer of thinking skills; attaining this goal requires systematic and explicit instruction. A key question in this respect is if, and to what extent, this training has to be built in the programming environment, and which part of it can better remain in the hands of the teacher. In other words, should we develop intelligent computer systems that can perform detailed analyses of pupils' skills and strategies, and subsequently provide adequate coaching and instruction? Or should the latter activities rather be performed by the teacher, while the major role of the computer is to create an environment that stimulates learners to practise their own knowledge and thinking skills [21]? An example of a system that focussed on eliciting the use of general thinking skills in a programming environment, is the Michael system. This system, developed at the Department of Computer Sciences of the University of Leuven in cooperation with our research group, explicitly stimulates the step-by-step construction and refinement of programs. It also offers several opportunities to free the user from the syntactical aspects of programming, so that he can fully concentrate on the more essential problem-solving and designing activities [22].

Finally, it seems to us that future research and developmental work should not be limited to learning environments in which programming is the core component. Indeed, since we do not consider the cognitive effects observed in this study as resulting from the programming activity as such, but rather from the entire learning environment in which it is embedded, we have no ground for attributing computer programming a monopoly position in teaching thinking skills. Consequently, it is plausible to assume that the investment of the same amount of time, effort and expertise, can produce equally effective learning environments in more "traditional" content domains, and this would facilitate the integration of the teaching of thinking skills within the regular curriculum. In fact; this implies a plea for approaching the teaching of thinking from a diversity of content areas. After all, the acquisition of generally applicable problem-solving skills requires that children are taught to apply them in a diversity of domains and problem situations. This can only be realized through instructional planning across subject-matter domains focussing on the elaboration of a common strategy for thinking-oriented instruction.

REFERENCES

1. L.B. Resnick, Learning in school and out, *Educational Researcher, 16*: 9, pp. 13-20, 1987.
2. D.N. Perkins and G. Salomon, Are cognitive skills context-bound? *Educational Researcher, 18*: 1, pp. 16-25, 1989.
3. S. Papert, *Mindstorms. Children, computers, and powerful ideas*, Basic Books, New York, 1980.
4. W. Feurzeig, P. Horowitz, and R.S. Nickerson, *Microcomputers in education*, (Report No. 4789), Bolt, Beranek and Newman, Cambridge, MA, 1981.
5. R.S. Nickerson, Computer programming as a vehicle for teaching thinking skills, *Thinking. The Journal of Philosophy for Children, 4*, pp. 42-48, 1983.
6. M.L. Gick and K.J. Holyoak, The cognitive basis of knowledge transfer, in *Transfer of learning. Contemporary research and applications*, (Educational Technology Series), S.M. Cormier and J.D. Hagman (eds.), Academic Press, San Diego, pp. 9-46, 1987.
7. M. Pressley, B.L. Snyder, and T. Cariglia-Bull, How can good strategy use be taught to children? Evaluation of six alternative approaches, in *Transfer of learning. Contemporary research and applications*, (Educational Technology Series), S.M. Cormier and J.D. Hagman (eds.), Academic Press, San Diego, pp. 81-120, 1987.
8. E. De Corte and L. Verschaffel, Effects of computer experience on children's thinking skills, *Journal of Structural Learning, 9*, pp. 161-174, 1986.
9. E. De Corte and L. Verschaffel, LOGO, a vehicle for thinking, in *New directions in mathematics education*, B. Greer and G. Mulhern (eds.), Routledge, London, pp. 63-81, 1989.
10. E. De Corte, L. Verschaffel, E. Hoedemaekers, H. Schrooten, and R. Indemans, *Acquiring planning skills in LOGO: An exploratory study with sixth graders*, paper presented at the annual conference of the International Association for Computing in Education, New Orleans, April, 1988.
11. E. De Corte, Towards powerful learning environments for the acquisition of problem-solving skills, *European Journal of Psychology of Education*, in press.

12. A. Collins, J.S. Brown, and S.E. Newman, Cognitive apprenticeship: Teaching the craft of reading, writing and mathematics, in *Cognition and instruction: Issues and agendas*, L.B. Resnick (ed.), Erlbaum, Hillsdale, NJ, in press.

13. S. McCoy Carver, *Transfer of LOGO debugging skill: Analysis, instruction and assessment*, unpublished doctoral dissertation, Department of Psychology, Carnegie-Mellon University, Pittsburg, PA, 1986.

14. J.D. Bransford and B.S. Stein, *The ideal problem solver*, Freeman, New York, 1984.

15. L. Sowder, Children's solutions of story problems, *Journal of Mathematical Behavior*, *7*, pp. 227-238, 1988.

16. R.E. Mayer, *Teaching and learning computer programming. Multiple research perspectives*, Erlbaum, Hillsdale, NJ, 1988.

17. G. Salomon and D.N. Perkins, Transfer of cognitive skills from programming: When and how? *Journal of Educational Computing Research*, *3*, pp. 149-169, 1987.

18. G. Salomon, D.N. Perkins, and T. Globerson, *Partners in cognition: Extending human intelligence with intelligent technologies*, invited address presented at the annual meeting of the American Educational Research Association, New Orleans, April, 1988.

19. L.B. Resnick, Toward a cognitive theory of instruction, in *Learning and motivation in the classroom*, S.G. Paris, G.M. Olson, and H.W. Stevenson (eds.), Erlbaum, Hillsdale, NJ, pp. 5-38, 1983.

20. E. De Corte, *Toward a theory of learning form instruction: The case of cognitive skills*, invited address presented at the First European Congress of Psychology, Amsterdam, The Netherlands, July, 1989.

21. M. Scardamalia, C. Bereiter, R.S. McLean, J. Swallow, and E. Woodruff, Computer-supported intentional learning environments, *Journal of Educational Computing Research*, *5*, pp. 51-68, 1989.

22. T. Tollenaere and H. Olivié, *An educational programming environment*, unpublished master's thesis, University of Leuven, Department of Computer Sciences, Leuven, 1988.

Corresponding address:
E. De Corte
Center for Instructional Psychology
University of Leuven
Vesaliusstraat 2
B-3000 Leuven
Belgium

Table 1. Overview of the activities in the LOGO course

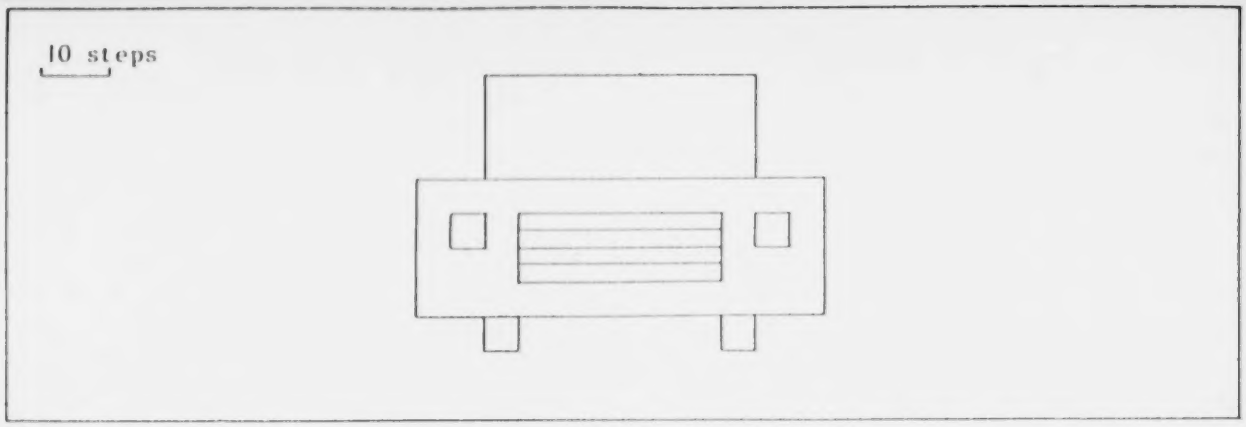
Unit no.	Activity
1	Acquaintance with the computer
2-4	Elementary primitives: FD/BK, RT/LT, PU/PD
5	REPEAT command (the list to be repeated only contains elementary primitives)
6-7	Procedures
8-9	Superprocedures
10	REPEAT command (the list to be repeated also contains procedures)
11	Demonstration of the entire programming strategy
12-14	Demonstration and practice of the seperate components of the programming strategy
15-16	Practice of the strategy with relatively simple given drawings
17-19	Practice of the strategy with relatively complex given drawings
20-22	Practice of the strategy with self chosen drawings
	First testsession for mastery tests
23-25	Practice of the strategy with given drawing containing circles
	Remedial instruction
	Second testsession for mastery tests

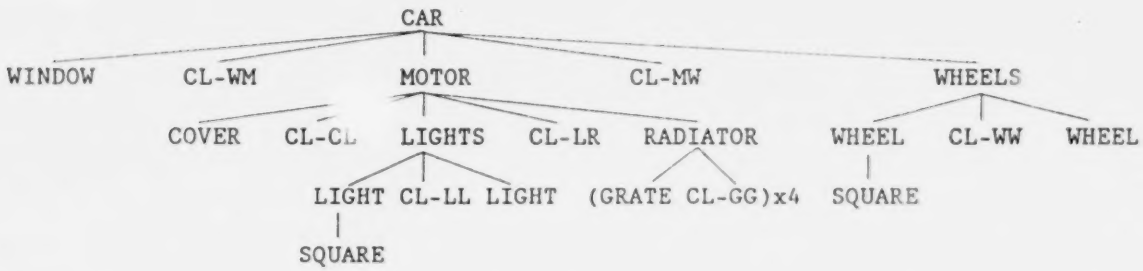
Table 2. Number of pupils in E1 and E2 who attained the mastery criterion on the tests for the three transfer conditions

Transfer conditions	Mastery tests	Results (n=24)	
		E1	E2
1. LOGO knowledge	LOGO knowledge	21	17
2. Thinking skills in LOGO	Planning 1	19	23
	Planning 2	20	19
	Executing/testing	20	22
3. Thinking skills in math	Planning 1	-	22
	Planning 2	-	10
	Executing/testing	-	22

Table 3. Comparison of E1 vs. C and E1 vs. E2 on the transfer tests (MANOVA)

Measures	Thinking skills	Results	
		E1 vs. C	E1 vs. E2
Mazes/pl1 Mazes/pl2 Error/pln	Planning	.249	>.500
Error/deb	Debugging	.001	.016
Block/dec Weekd/dec	Decomposition	<.001	.002
Weekd/ext Boxes/ext	Use of external representation	<.001	<.001





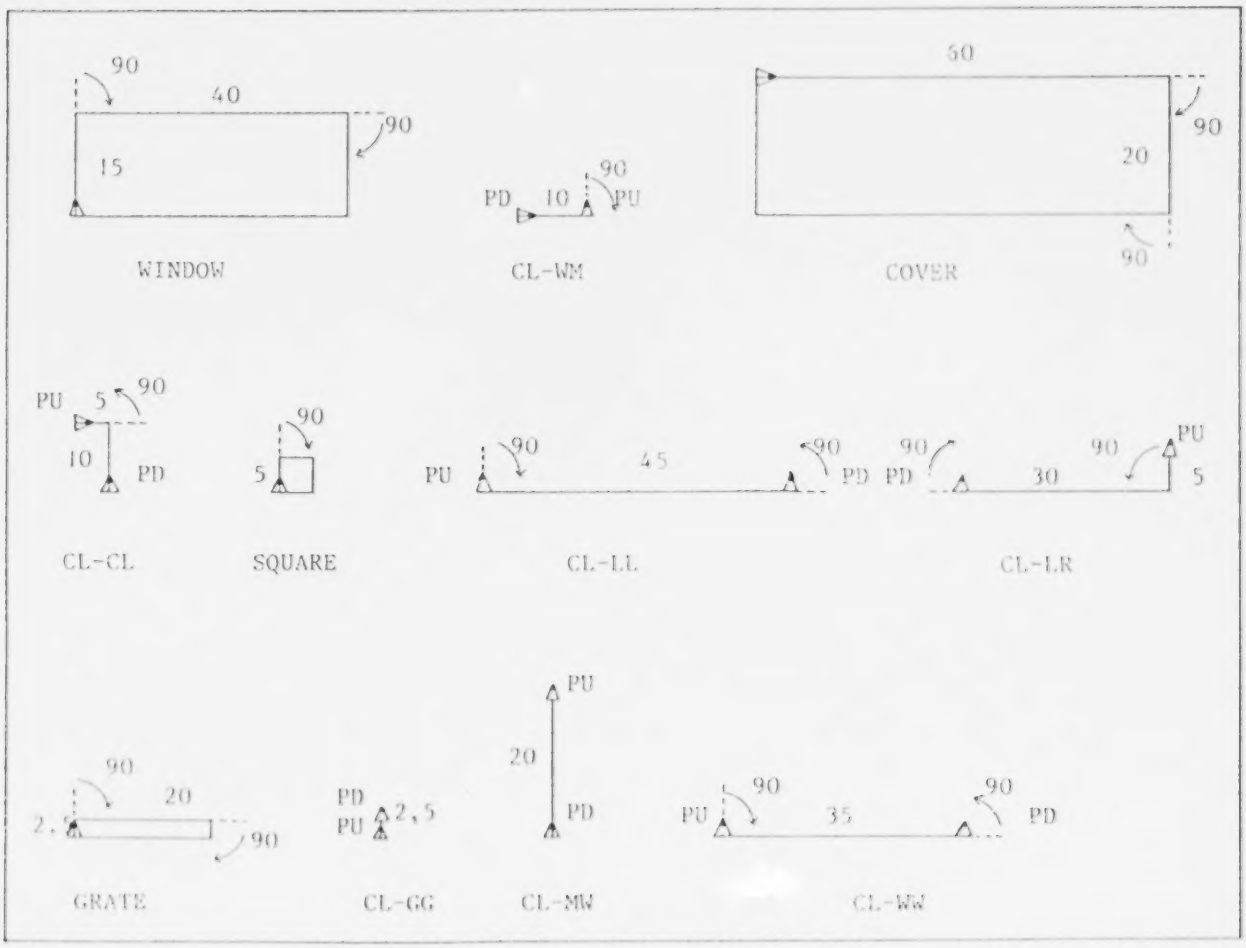


Figure 1. Plan for writing a LOGO program for drawing a car

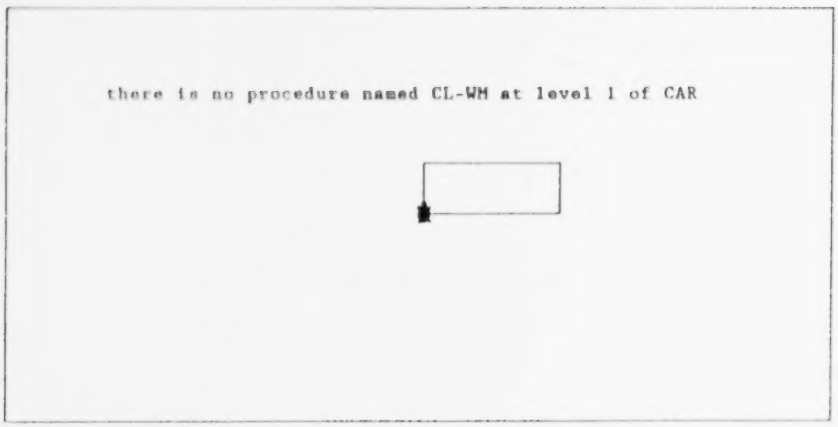


Figure 2. Screen effect after coding the procedure WINDOW

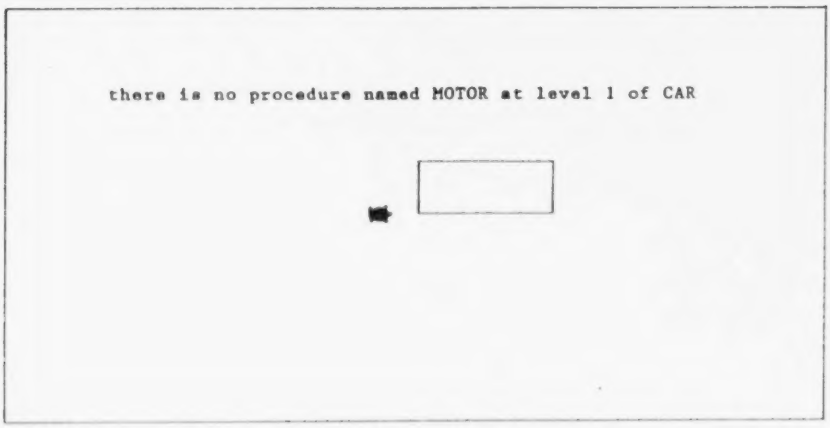


Figure 3. Screen effect after coding the procedure CL-WM

The carpet in our living room which measures 5m by 4,40m, has to be renewed. The price of the carpet is 350 per m². The craftsman charges 600 fr per hour and works 3 hours on the job. What will be the total cost for the renewal of the carpet?

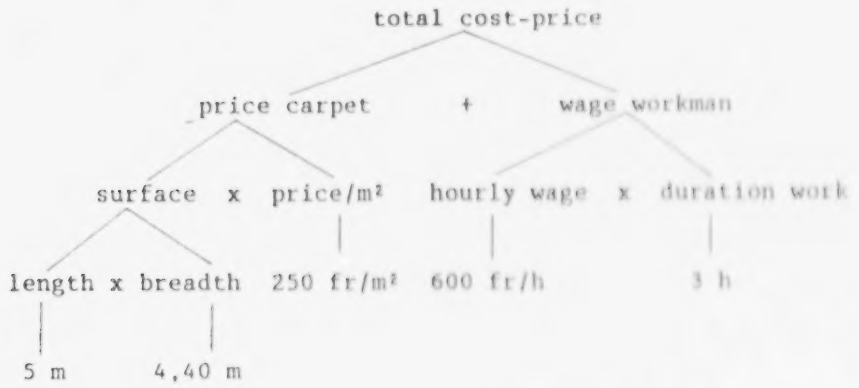
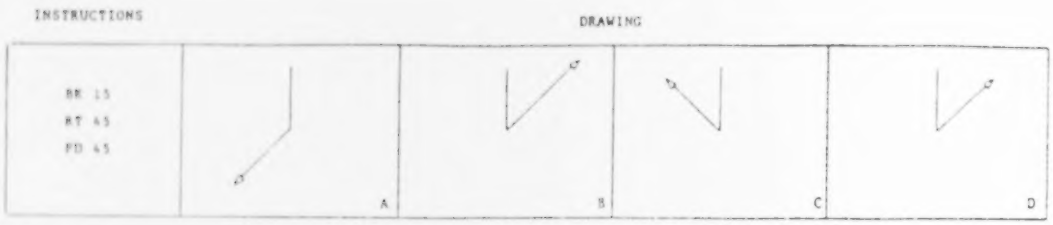
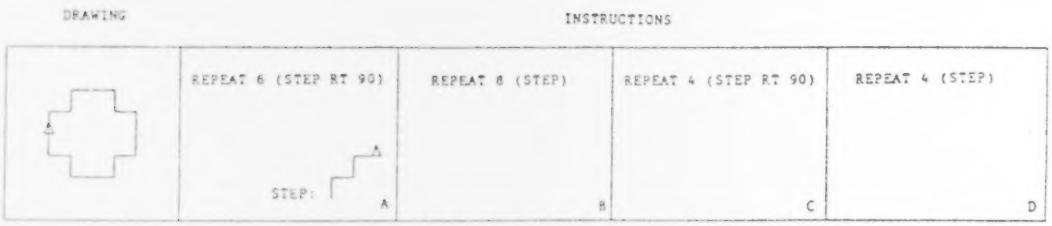


Figure 4. Plan for solving a multi-step word problem

(a) Mark the drawing that the turtle will make when given the following instructions



(b) Mark the instructions that the turtle recieved in order to make the following drawing



(c) When the turtle executes the following program, it draws the parts of the figure in a well-defined sequence. The numbers in the drawing indicate the sequence in which the turtle draws the parts. Mark the drawing with the correct sequence.

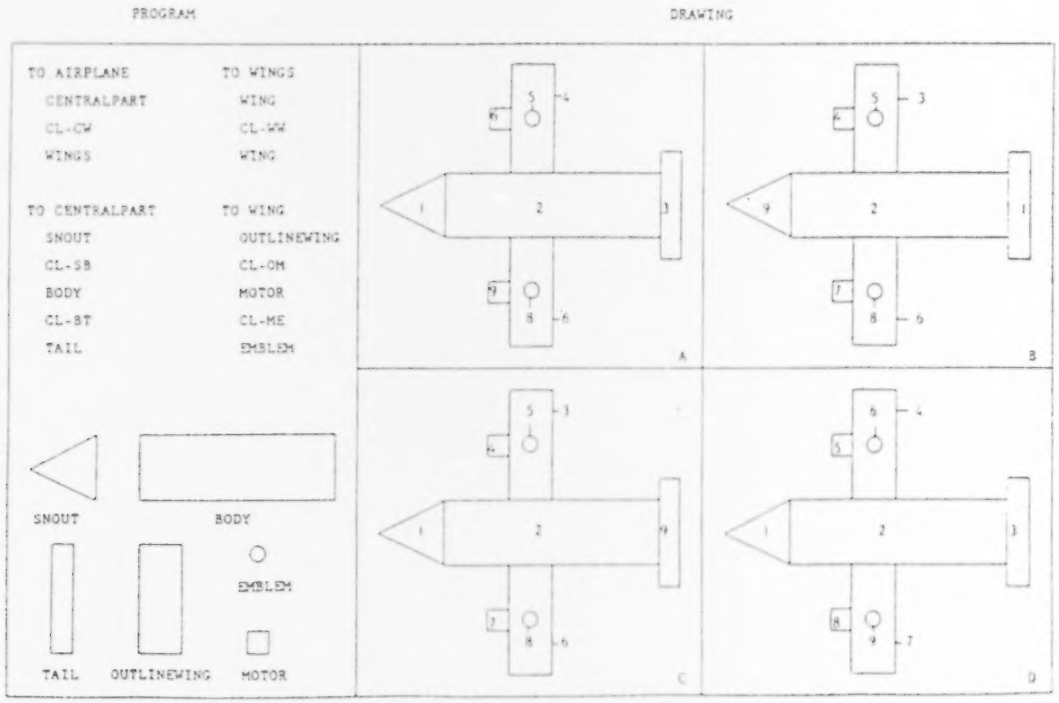
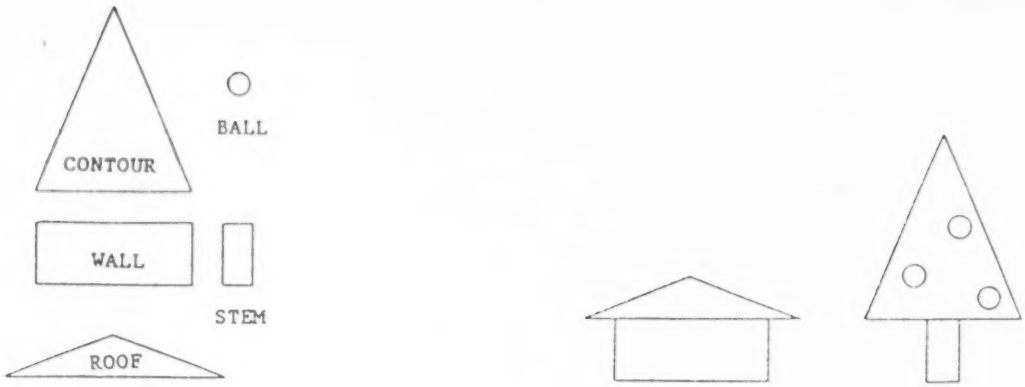
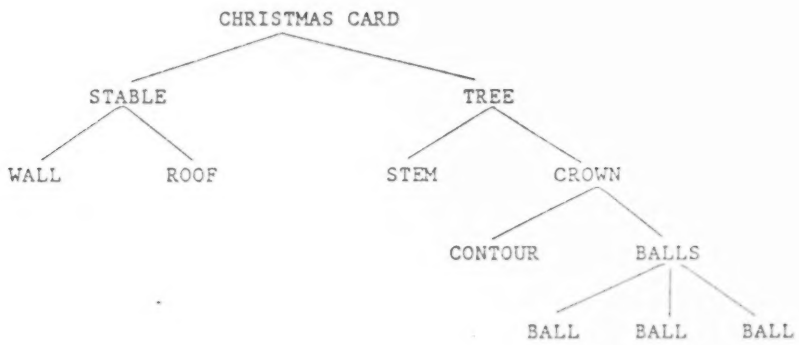


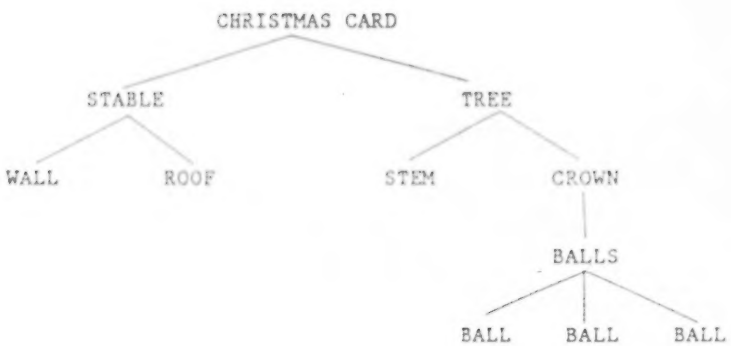
Figure 5. Example of an item form the LOGO knowledge test about the elementary primitives (a), the REPEAT command (b), and the flow of control (c)



A



B



The best tree diagram is:

Figure 6. Example of an item from the first planning test

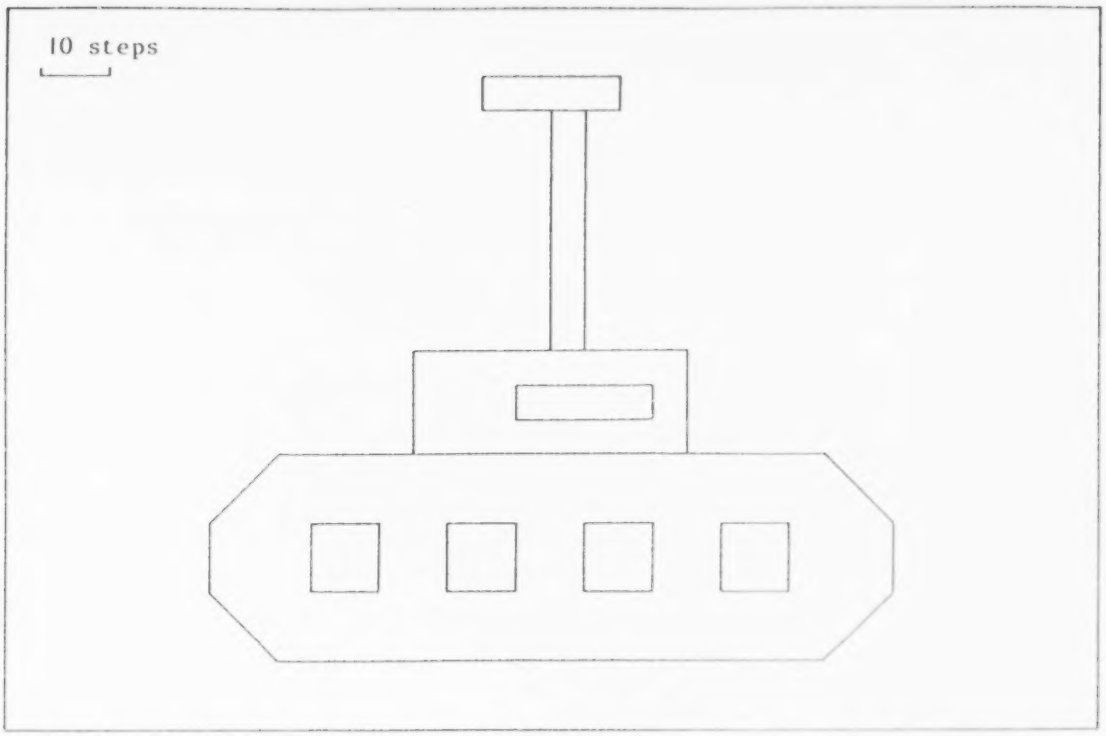
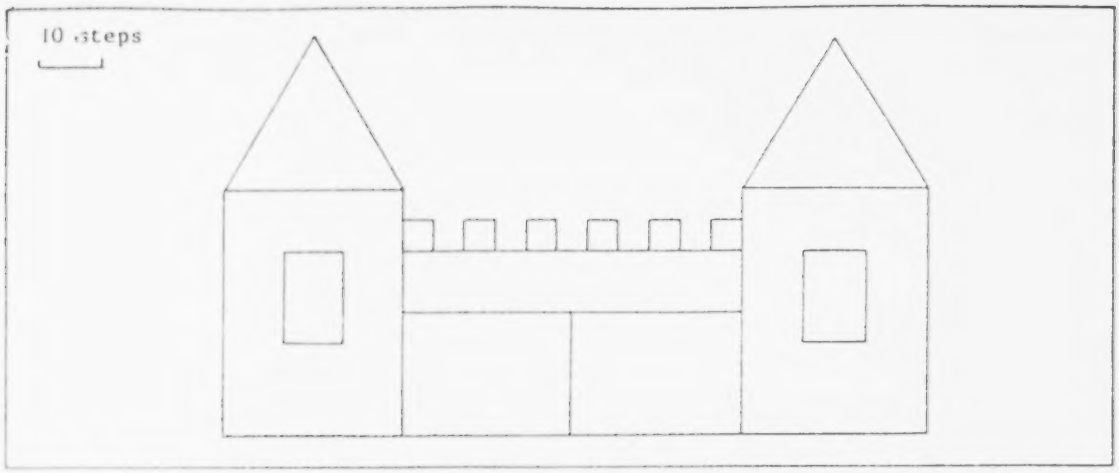
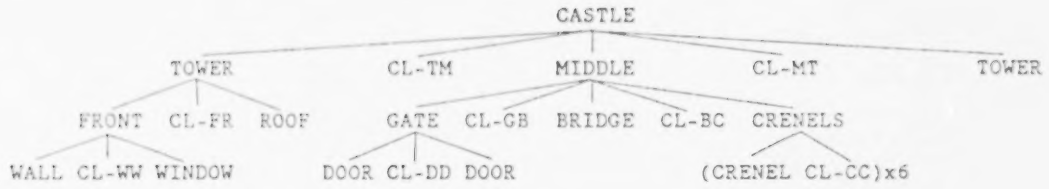


Figure 7. Drawing of a submarine for which the children had to draw a tree diagram in the second planning test





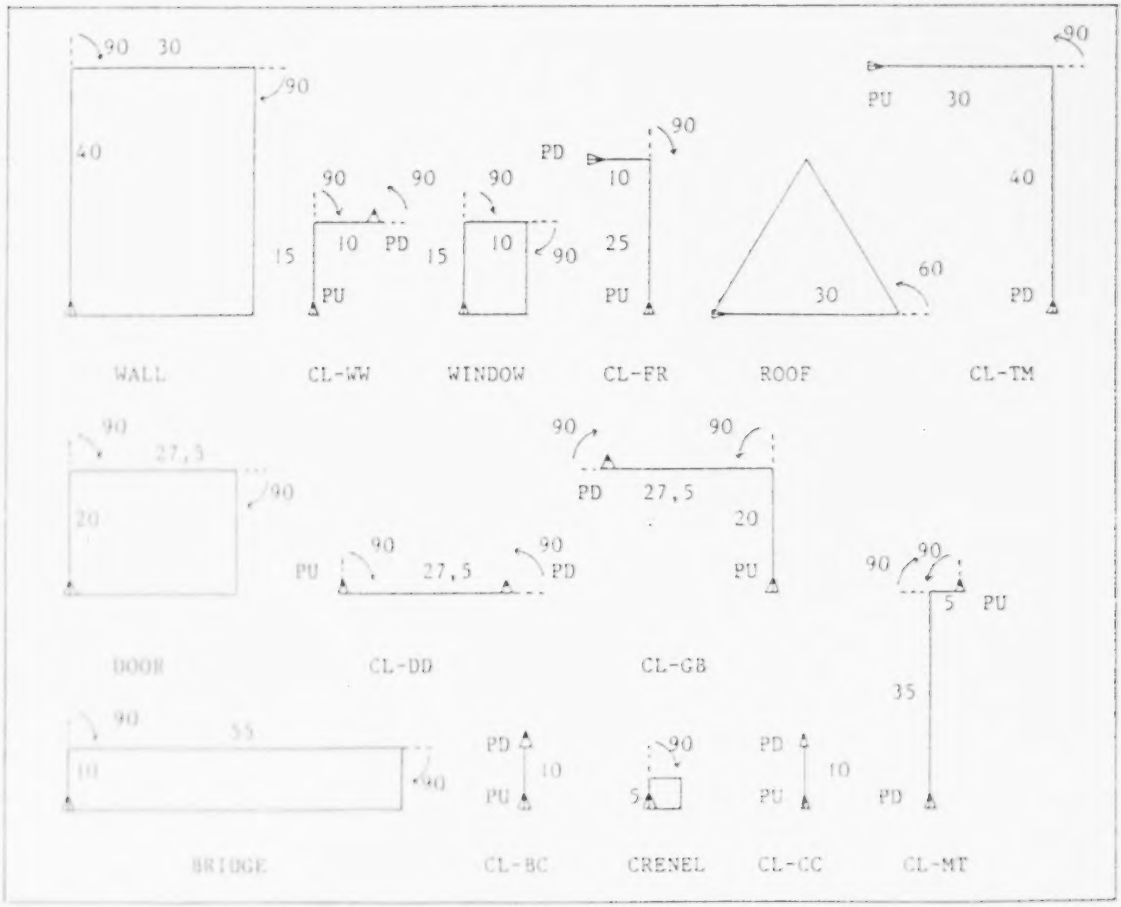


Figure 8. Drawing of a castle, together with a faulty plan, given as starting point for the executing-and-testing phase

In this maze you see three salads: a yellow, a green and a brown one. In the middle of the maze you see a red figure. This is a hungry beatle. He now wants to eat the three salads!
You can steer the beatle as follows: if you push an arrow-key, the beatle is set in the direction of the arrow; if you push the space-bar, the beatle moves one step forward. As soon as the beatle reaches the salad, it is eaten.
Mind this! The beatle is very lazy; it wants to move as few steps as possible to eat the three salads. Therefore, the covered distance (this is the distance from the starting point up to the last salad) should be as short as possible. The time is of no importance.

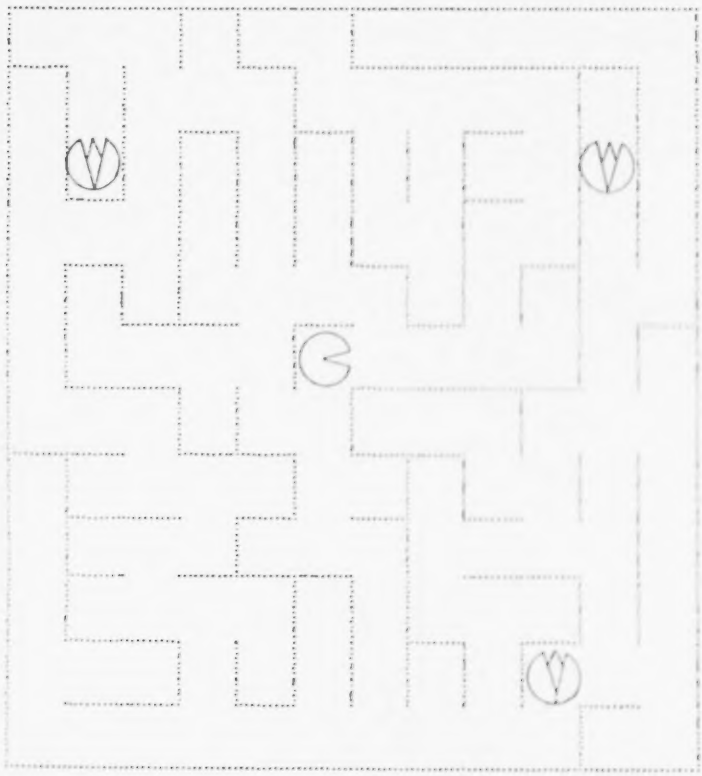
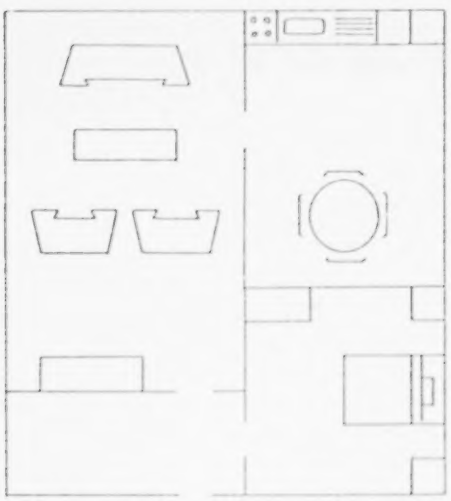


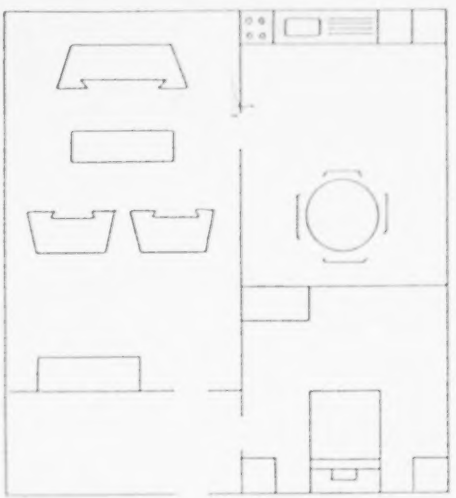
Figure 9. Example of an item from the mazes test

Mrs. Fisher was moving into a new apartment with the help of two movers. She asked them to arrange the furniture in het apartment and gave them a list of directions. The movers followed the instructions perfectly; however there was a bug in the directions; consequently the furniture was not arranged correctly. Picture A shows how Mrs. Fisher wanted her apartment to look like; picture B how the furniture was really arranged by the movers. Try to find the bug in Mrs. Fisher's directions and correct it.

Picture A



Picture B



These are Mrs. Fisher's directions.

In the kitchen:
- Put the electric cooker at the left of the dresser.
- Put the automatic dishwasher at the other side of the dresser.
- Put the refrigerator at the right of the automatic dishwasher.
- Put the table in the south-eastern corner of the kitchen.
- Put the chairs around the table.

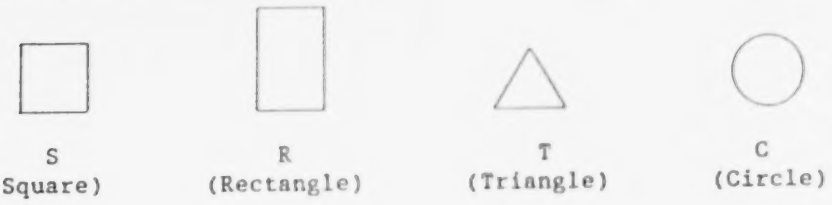
In the living room:
- Put the sofa against the northern wall.
- Put the t seats facing the sofa.
- Put the table between the sofa and the seats.
- Put the cabinet against the southern wall.

In the bedroom:
- Put the head of the bed against the southern wall.
- Put a pedestal on each side of the bed.
- Put the wardrobe against the wall of the kitchen.

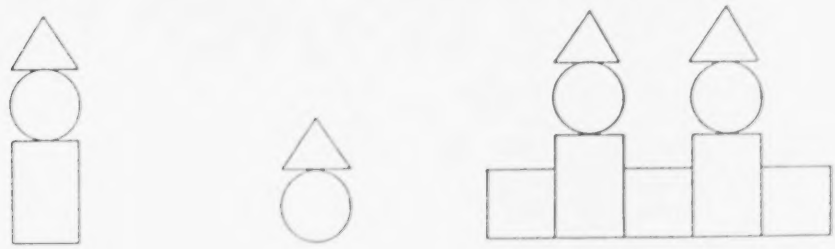
Figure 10. Example of an item from the error detection test

Describe the complex figures by indicating which elements they contain. You have to use either the single elements (the blocks S, R, T, C) or the other complex figures. It is important that you use as few elements as possible; therefore you have to choose each time the biggest available figure. In enumerating the elements, you work from left to right and from bottom to top.

Here is an example:



S (Square) R (Rectangle) T (Triangle) C (Circle)



1 2 3

Figure number	Solution
1	R, 2
2	C, D
3	V, 1, V, 1, V

Figure 11. Example of an item from the blocks test

Figure 12. Two examples of decomposition and external representation of an item from the weekdays test

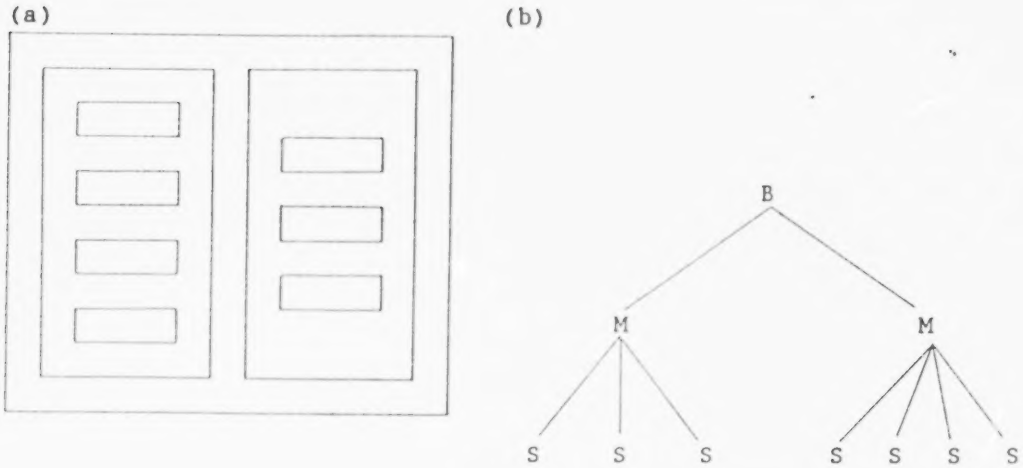


Figure 13. Two examples of external representation of an item from the boxes test